ELSEVIER
Discrete Applied Mathematics 72 (1997) 199–207
DISCRETE APPLIED MATHEMATICS

A d-move local permutation routing for the d-cube

Frank K. Hwang[a,*], Y.C. Yao[b], Miltos D. Grammatikakis[c]

[a] *AT&T Bell Laboratories, 600 Mountain Avenue, P.O. Box 636, Murray Hill, NJ 07974-0636, USA*
[b] *Department of Statistics, Colorado State University, Fort Collins, CO 80523, USA*
[c] *Ecole Normale Superieure de Lyon, LIP Lyon 69364, France*

Received 28 December 1992; revised 19 August 1994

Abstract

Optimal packet routing algorithms for all binary d-cubes of dimension $d \leqslant 7$ are presented. The algorithms given are synchronous, offer distributed control, and assume d-port, multiaccepting communication. While the previous best known packet routing algorithm [3] on the 7-cube takes 11 time-units, our algorithm has reduced the worst-case time complexity to the minimum possible of 7 units. We also give an optimal routing algorithm for the ternary 4-cube.

Key words: d-cube; interconnection networks; Local permutation routing; Maximal checking; Packet routing

1. Introduction

The state of the art in parallel systems includes multicomputer architectures such as hypercubes (Ncube, Intel's IPSC, Connection Machine), and multiprocessors such as the butterfly [2, 8]. The usual communication mode is either *packet* or *circuit switching*. In circuit (packet) switching data are transferred through the interconnection network with (without) a physical connection path between the source and the destination. Packet routing forms the kernel of many parallel processing applications, such as sorting, computing multidimensional FFTs, matrix transposition, and divide and conquer strategies [4, 5, 7, 9]. The scope of this study is *packet routing on hypercubes*.

We consider a network whose topology is a binary d-dimensional cube with two directed edges, one in each direction, between each pair of adjacent nodes. In the beginning, each node contains a message with a destination, and the $n = 2^d$ destinations are all distinct (permutation routing). The modes are *synchronous* and the

*Corresponding author.

PII S0166-218X(96)00019-4

routing is divided into "moves." At each move a message either stays put, or moves to an adjacent node if the edge is free. In other words, at any move a message occupies either a node (if staying put), or an edge. As in the *multiple-accepting PE* scheme studied by Abraham and Padmanabhan [1], we allow multiple messages to occupy the same node. However, only one message can occupy an edge at any move. When several messages compete for the same edge (called a conflict), we assume that there is a register to decide which one occupies the edge. Finally, we assume that each node is capable of doing *d-port communication*, i.e. it can simultaneously receive and transmit up to d messages as long as no two are claiming the same edge.

For $2 \leqslant d \leqslant 7$ we give in Section 2 an algorithm which can complete an arbitrary permutation routing in d moves, the worst-case minimum number required for general permutations. The routing is *local* in the sense that at each move, the transmissions of the messages at a node depend only on the node and the destinations of these messages (but independent of what happens at other nodes). It is an interesting open problem whether a d-move local algorithm for permutation routing exists for arbitrary d.

We also consider an extension from binary cubes to t-nary cubes. However, the routing is much harder and we only have some limited success for small t and d. In particular, the case of $t = 3$ and $d = 4$ is discussed in Section 3 and some remarks are given in Section 4.

2. Binary d-cubes

We first state some basic but useful lemmas and definitions.

Lemma 1. A conflict cannot occur at either the first move or the last move.

Proof. At the first move no two messages can have the same starting node. At the last move no two messages share the same destination. Hence no conflict can occur at these two moves. □

Corollary 2. There exists a 2-move local permutation routing for the 2-cube.

We assume that each of the 2^d nodes of the d-cube is labelled by a binary d-vector. We also identify a message by its destination, i.e. a d-vector. We say bit i of a message is *correct* if its current node binary representation agrees with its destination in bit i. For example, consider $d = 4$ and two messages with respective destinations (0, 1, 0, 1) and (0, 1, 0, 0) at the node (0, 0, 0, 0). The first message has correct bits 1 and 3, while the second message has correct bits 1, 3 and 4. By *correcting bit i*, we mean a move to make bit i correct. Thus, when correcting bit i, a message stays put if bit i is correct, or else moves to the adjacent node with the destination bit i. In the above example, "correcting bit 4" will result in the first message moving to the node (0, 0, 0, 1) and the

second message staying put. For a subset P of $\{1, 2, \ldots, d\}$, we say a message is of pattern P if all bits in P are correct (bits not in P can either be correct or incorrect). In the above example, the first message is of pattern P for every subset P of $\{1, 3\}$ since its destination (0, 1, 0, 1) agrees with the (current) node (0, 0, 0, 0) in bits 1 and 3. Similarly, the second message is of pattern P for every subset P of $\{1, 3, 4\}$. For a given pattern P, we say a message is *maximal* (of P) if it is of pattern P and every bit not in P is incorrect. In the above example, with respect to pattern $\{1, 3\}$, the first message is maximal but not the second.

For a given k ($1 \leqslant k < d$), suppose all the messages have at least k correct bits. By *making the maximal correction of size k*, we mean (i) for every pattern P of size (cardinality) k, we correct a specified bit (not in P) for the maximal message of pattern P, and (ii) those messages having more than k correct bits stay put. Thus, after making the maximal correction of size k, if no conflict arises, then all the messages will have at least $k + 1$ correct bits. In the above example, after making the maximal correction of size 2, the second message stays put while the first message goes to either the node (0, 1, 0, 0) (if we correct bit 2) or the node (0, 0, 0, 1) (if we correct bit 4).

Lemma 3. At each node there exists at most one maximal message for each pattern.

Proof. Two maximal messages at a node of the same pattern imply that they have the same destination, which is impossible. □

Theorem 4. There exists a d-move local permutation routing for the binary d-cube for $3 \leqslant d \leqslant 7$.

Proof. $d = 3$: At the first move we correct bit 1. No conflict can occur due to Lemma 1. Furthermore, at the beginning of the second move, every message is of pattern $\{1\}$. At the second move we make the maximal correction of size 1; correct bit 2 of the maximal message of pattern $\{1\}$ (there exists no maximal message of pattern $\{2\}$ or $\{3\}$). By Lemma 3 at most one message is routed at each node, thus no conflict can occur. No conflict can occur at the third move by Lemma 1.

$d = 4$: At the first move we correct bit 1. No conflict can occur due to Lemma 1. At the second move we correct bit 2 if possible (which means that the dimension-2 edge has not been occupied by another message at move 2). If it is not possible, we correct bit 3. Since each node has at most two messages at the beginning of move 2, either bit 2 or bit 3 is corrected for every message and no conflict can occur. Thus, after the second move, every message is of pattern $\{1, 2\}$ or $\{1, 3\}$. At the third move we make the maximal correction of size 2 and we correct bit 3 (2) for the maximal message of pattern $\{1, 2\}$ ($\{1, 3\}$). No conflict can occur by Lemma 3. Clearly, after the third move, every message has at least 3 correct bits. Finally, no conflict can occur at move 4 by Lemma 1.

$d = 5$: At the first move if bit 1 is incorrect, correct bit 1; otherwise, correct bit 2. At the beginning of the second move, consider a general node $N_5 = (i_1, i_2, i_3, i_4, i_5)$.

Interpreting $a + b$ as $a + b$ (mod 2), the messages at that node can only come from nodes $(i_1 + 1, i_2, i_3, i_4, i_5)$, $(i_1, i_2, i_3, i_4, i_5)$ and $(i_1, i_2 + 1, i_3, i_4, i_5)$. (Note that the messages from the last two nodes already have correct bits 1 and 2.) Correct bit 2 or 3 or 4 for the respective message whenever it exists. Since we route at most three messages from N_5, each along a different dimension (edge), no conflict can occur. At the beginning of the third move, it is easily verified that all messages are of pattern $\{1, 2\}$. Clearly, at each node at most two messages can have destinations disagreeing with the node in both bits 3 and 4, hence not of pattern $\{1, 2, 3\}$ or $\{1, 2, 4\}$. Correct bit 3 or 4 for these messages whenever they exist. At the beginning of the fourth move, all messages are of pattern $\{1, 2, 3\}$ or $\{1, 2, 4\}$. We make the maximal correction of size 3 and correct bit 4 (3) for the maximal message of pattern $\{1, 2, 3\}$ ($\{1, 2, 4\}$). No conflict can occur due to Lemma 3. After the fourth move, all messages have at least 4 correct bits.

$d = 6$: The first three moves are the same as in the 5-cube case. (It should be remarked that it is still true, though a bit less obvious than in the case $d = 5$, that at the beginning of the third move at most two messages can have incorrect bits 3 and 4. This is because at every node all messages except at most two had either bit 3 or bit 4 corrected at the second move.) At the beginning of the fourth move, all messages have the patterns either $\{1, 2, 3\}$ or $\{1, 2, 4\}$. Among the messages of pattern $\{1, 2, 3\}$ ($\{1, 2, 4\}$) at the node $N_6 = (i_1, i_2, i_3, i_4, i_5, i_6)$, at most two can have incorrect bits 4 and 5 (3 and 6). We correct bit 4 (3) for the first message and bit 5 (6) for the second. All other messages stay put. Since the corrected bits are all distinct, no conflict can occur. Thus, after the fourth move, all messages are of pattern $\{1, 2, 3, 4\}$, $\{1, 2, 3, 5\}$ or $\{1, 2, 4, 6\}$. At the fifth move we make the maximal correction of maximal correction of size 4; for the maximal message of pattern $\{1, 2, 3, 4\}$ ($\{1, 2, 3, 5\}$, $\{1, 2, 4, 6\}$) we correct bit 5 (4, 3, respectively). Since the corrected bits are all distinct, no conflict can occur. After the fifth move, all messages have at least five correct bits.

$d = 7$: We will describe the first four moves later. Assume for now that at the beginning of the fifth move, all messages are of pattern $\{1, 2, 3, 4\}$, $\{1, 2, 3, 5\}$ or $\{1, 2, 4, 6\}$. Among the messages of pattern $\{1, 2, 4, 6\}$ ($\{1, 2, 3, 5\}$) at the node $N_7 = (i_1, i_2, i_3, i_4, i_5, i_6, i_7)$, at most two can have incorrect bits 3 and 5 (4 and 6). Correct bit 3 (4) for the first message and bit 5 (6) for the second. For the maximal message of pattern $\{1, 2, 3, 4\}$, correct bit 7. Since all corrected bits are distinct, no conflict can occur. After the fifth move, all messages are of pattern $\{1, 2, 3, 4, 5\}$, $\{1, 2, 3, 4, 6\}$, $\{1, 2, 3, 4, 7\}$, $\{1, 2, 3, 5, 6\}$ or $\{1, 2, 4, 5, 6\}$. At the sixth move we make the maximal correction of size 5; for the maximal message of pattern $\{1, 2, 3, 4, 5\}$ ($\{1, 2, 3, 4, 6\}$, $\{1, 2, 3, 4, 7\}$, $\{1, 2, 3, 5, 6\}$, $\{1, 2, 4, 5, 6\}$), we correct bit 6 (7, 5, 4, 3, respectively). Since all corrected bits are distinct, no conflict can occur. After six moves, all messages have at least six correct bits.

The first two moves are the same as in the 5-cube and 6-cube cases. The third move is complicated. Note that at the beginning of the third move, N_7 has at most six messages of which no more than four can have an incorrect bit 3 (4). Among the messages at N_7 (at the beginning of the third move), let A be the set of those

messages having destinations with number $i_3 + 1$ ($i_4 + 1$) in bit 3 (4), B the set of those messages having destinations with number i_3 ($i_4 + 1$, $i_5 + 1$) in bit 3 (4, 5), C the set of those messages having destinations with number i_4 ($i_3 + 1$, $i_6 + 1$) in bit 4 (3, 6). Thus, $|A| \leqslant 2$, $|A| + |B| = |A \cup B| \leqslant 4$, $|A| + |C| = |A \cup C| \leqslant 4$, $|A| + |B| + |C| = |A \cup B \cup C| \leqslant 6$. The detailed description of the third move is given below, which depends on the values of $|A|$, $|B|$ and $|C|$ in such a way that neither of the messages in A stays put and at most one of the messages in $B(C)$ stays put. (Note that all messages have correct bits 1 and 2.)

(i) $|A| = 0$, $|B| = 4$, $|C| \leqslant 2$: Note that the four messages in B are the maximal messages of pattern $\{1, 2, 3\}$, $\{1, 2, 3, 6\}$, $\{1, 2, 3, 7\}$ and $\{1, 2, 3, 6, 7\}$. Correct bit 5, 7, 6 or 4 for these messages, respectively. Correct bit 3 for one of the messages in C.

(i') $|A| = 0$, $|B| \leqslant 2$, $|C| = 4$: Same as in (i) with B and C interchanged. Namely, correct bit 4 for one of the messages in B, and correct bit 6, 7, 5 or 3, respectively, for the four maximal messages of pattern $\{1, 2, 4\}$, $\{1, 2, 4, 5\}$, $\{1, 2, 4, 7\}$ and $\{1, 2, 4, 5, 7\}$ in C.

(ii) $|A| \leqslant 1$, $|B| = 3$, $|C| \leqslant 2$: Correct bit 4 for the message in A (if it exists). Correct bit 3 for one of the messages in C. Correct bit 4 for the message in A (if it exists). Correct bit 3 for one of the messages in C. Correct bit 5 (6, 7) for the maximal message of pattern $\{1, 2, 3\}$ ($\{1, 2, 3, 7\}$, $\{1, 2, 3, 6\}$, respectively) in B (if it exists).

(ii') $|A| \leqslant 1$, $|B| \leqslant 2$, $|C| = 3$: Same as in (ii) with B and C interchanged. That is, correct bit 3 for the message in A (if it exists), correct bit 4 for one of the messages in B, and correct bit 6 (5, 7) for the maximal message of pattern $\{1, 2, 4\}$ ($\{1, 2, 4, 7\}$, $\{1, 2, 4, 5\}$, respectively) in C (if it exists).

(iii) $|A| = 0$, $|B| = |C| = 3$: Correct bit 4 or 5 for two of the messages in B, and correct bit 3 or 6 for two of the messages in C.

(iv) $|A| \leqslant 2$, $|B| \leqslant 2$, $|C| \leqslant 2$: Correct bit 3 or 4 for the messages in A, correct bit 5 (6) for one of the messages in B (C).

Since the third move always corrects bit 3 or 4 for the messages in A, all messages are of pattern $\{1, 2, 3\}$ or $\{1, 2, 4\}$ afterwards. We claim that among the messages of pattern $\{1, 2, 3\}$ ($\{1, 2, 4\}$) at each node, at most two can have incorrect bits 4 and 5 (3 and 6). Assuming the claim for now, then the fourth move is to correct bit 4 or 5 (3 or 6) for these messages, so that all messages are of pattern $\{1, 2, 3, 4\}$, $\{1, 2, 3, 5\}$ or $\{1, 2, 4, 6\}$ at the beginning of the fifth move.

Finally, it remains to establish the claim. Without loss of generality, we consider node $\boldsymbol{e}_0 = (0, 0, 0, 0, 0, 0, 0)$. We shall only consider the case of pattern $\{1, 2, 3\}$ as the other case can be treated similarly. Namely, we need to show $|B^*| \leqslant 2$ where B^* is the set of those messages at node $\boldsymbol{e}_0$ (after the third move) having destinations with 0 in bits 1, 2 and 3 and 1 in bits 4 and 5.

Denote by $\boldsymbol{e}_i$ the node $(0, 0, \ldots, 0, 1, 0, \ldots, 0)$ with the only "1" appearing in the ith component, $i = 1, 2, \ldots, d$. For $i = 0, 1, \ldots, d$, let B_i be the set of those messages at node the $\boldsymbol{e}_i$ (at the beginning of the third move) having correct bits 1, 2 and 3 and incorrect bits 4 and 5.

Table 1

	e_0	e_3	e_4	e_5	e_6	e_7
(i)	0010x1x	0000x1x	0000100	0001011	0001100	0001100
(i')	00011xx	0000010	00001xx	0010010	0010101	0010010
(ii)	0010x1x 0001100	0000x1x	0010xxx	0001011	0001100	0001100
(ii')	00011xx 0010010	0001xxx	00001xx	0010010	0010101	0010010
(iii)	00011xx 0010x1x	0000x1x	00001xx	00010xx	0010x0x	
(iv)	00011xx 0010x1x	0001xxx	0010xxx	00010xx	0010x0x	

Table 1 lists the possible message(s) at e_0 moved from e_i (column) at the third move using a certain rule (row) (x means an unspecified bit).

It is easily seen that any message in B^*, i.e. any message of the type 00011xx must come from e_0, e_3, e_6 or e_7. Also, at most one message in B^* can come from each of them. Furthermore, the message coming from e_6 or e_7 is the same one, i.e. 0001100. So $e_6 \cup e_7$ can contribute at most one message to B^*. Hence we will conclude that $|B^*| \leqslant 2$ if it can be shown that not all of e_0, e_3 and $e_6 \cup e_7$ can contribute a message to B^*.

Note that the moving of the message 0001100 from e_6 (e_7) to e_0 occurs either under rule (i) or under rule (ii). In either case $|B_6|$ ($|B_7|$) is at least 3. Furthermore, these messages in B_6 and B_7 must be of the type 00011xx. Now suppose e_0, e_3 and $e_6 \cup e_7$ all contribute a message to B^*. Denote by m_0 (m_3) the message coming from e_0 (e_3). Clearly, all messages in $\{m_0, m_3\} \cup B_6 \cup B_7$ are of the type 00011xx, so that

$$|\{m_0, m_3\} \cup B_6 \cup B_7| \geqslant |B_6 \cup B_7| + 2 \geqslant 3 + 2 = 5,$$

which is impossible since there are only four distinct messages of the type 0001xx. Therefore $|B^*| \leqslant 2$, completing the proof. □

3. t-nary d-cubes

It is straightforward to extend our results for binary 2-cube and 3-cube to similar size ternary cube. However, the extension to the ternary 4-cube is already nontrivial. For convenience, we still refer to the components in a d-vector as "bits."

Theorem 5. There exists a 4-move local permutation routing for the ternary 4-cube.

Proof. The first move of the routing algorithm to be given is complicated. Later, we will describe the first two moves and show that every message after two moves is of

pattern either $\{1, 3\}$ or $\{2, 4\}$. Assuming this fact for now, the third move is to make the maximal correction of size 2, to be described as follows. For a general node (i, j, k, l) with $i, j, k, l = 0, 1, 2$, at the beginning of the third move, there can be at most four maximal messages of pattern $\{1, 3\}$ with destinations $(i, j+1, k, l+1)$, $(i, j+2, k, l+2)$, $(i, j+1, k, l+2)$ and $(i, j+2, k, l+1)$, for which we correct bits 2, 2, 4 and 4, respectively where $a+b$ is interpreted as $a+b$ (mod 3). Similarly, there can be at most four maximal messages of pattern $\{2, 4\}$ with destinations $(i+1, j, k+1, l)$, $(i+2, j, k+2, l)$, $(i+1, j, k+2, l)$, and $(i+2, j, k+1, l)$, for which we correct bits 1, 1, 3 and 3, respectively. Clearly, no conflict will arise at the third move. Thus, after the third move, every message has at least three correct bits. The fourth move is simply to correct the only incorrect bit for each message, which results in no conflict by a natural extension of Lemma 1 to t-nary d-cubes.

We now describe the first move. Denote the 3^4 nodes by (i, j, k, l), $i, j, k, l = 0, 1, 2$. Denote by (i_1, j_1, k_1, l_1) the destination of the message at (i, j, k, l). Thus $i_1 = i_1(i, j, k, l)$ is a function of i, j, k and l. First consider those messages with at least one correct bit. If $i_1 = i$ (i.e. $i_1(i, j, k, l) = i$), correct bit 3; if $j_1 = j$, correct bit 4; if $k_1 = k$, correct bit 1; if $l_1 = l$, correct bit 2. (If $i_1 = i$ and $j_1 = j$, then we may correct either bit 3 or 4.) In other words, if a message has at least one correct bit before the move, then it will have correct bits $\{1, 3\}$ or $\{2, 4\}$ after the move, Next, consider those messages with no correct bit.

(i) If $(i, j) = (0, 1)$, $(1, 2)$ or $(2, 0)$, correct bit 2;

(ii) If $(i, j) = (1, 0)$, $(2, 1)$ or $(0, 2)$, correct bit 1;

(iii) For the case $i = j$ and $(l, l_1) = (0, 1)$, $(1, 2)$ or $(2, 0)$, correct bit 4. For the case $i = j$ and $(l, l_1) = (1, 0)$, $(2, 1)$ or $(0, 2)$, if $(j, j_1) = (0, 1)$, $(1, 2)$ or $(2, 0)$, correct bit 2; if $(j, j_1) = (1, 0)$, $(2, 1)$ or $(0, 2)$, correct bit 3.

The second move is simply to correct those messages with exactly one correct bit in such a way that they will be of pattern $\{1, 3\}$ or $\{2, 4\}$ after corrections. It suffices to show that these messages can cause no conflict at the second move since messages with at least two correct bits stay put at the second move.

Case 1: Fix a node (i, j, k, l) with $i = j$. We shall only consider $i = j = 0$. The other two cases can be treated similarly. Those messages with one correct bit must have come from some of the nodes $(1, 0, k, l)$, $(0, 1, k, l)$, $(0, 0, k, F(l))$, $(0, 0, k', l)$ and $(0, 0, k'', l)$ where $F(0) = 2$, $F(1) = 0$ and $F(2) = 1$, and $\{k', k''\} = \{0, 1, 2\} \backslash \{k\}$. For messages from the first three nodes (if they exist) correct bits 3, 4 and 2, respectively. For messages from the last two nodes (if they exist) correct bit 1. In case both messages from the last two nodes exist, denote their destinations by $(i_1^*, j_1^*, k_1^*, l_1^*)$ and $(i_2^*, j_2^*, k_2^*, l_2^*)$. Clearly, $k_1^* = k_2^* = k$. By the description of the first move, we have $l_1^* = l_2^* = F(l)$ and $j_1^* = j_2^* = 2$. Thus $i_1^* \neq i_2^*$, and no conflict will occur when correcting bit 1 of both messages.

Case 2: Fix a node (i, j, k, l) with $(i, j) = (0, 1)$, $(1, 2)$ or $(2, 0)$. We shall consider only $(i, j) = (0, 1)$, since the other two cases are similar. At the node $(0, 1, k, l)$, those messages with one correct bit must have come from the nodes $(2, 1, k, l)$ and $(0, 0, k, l)$. We correct bit 3 of the first message and bit 4 of the second.

Case 3: Fix a node (i, j, k, l) with $(i, j) = (1, 0), (2, 1)$ or $(0, 2)$. We treat $(i, j) = (1, 0)$ only. At the node $(1, 0, k, l)$ the only message with one correct bit must have come from $(1, 2, k, l)$. We correct bit 4 of this message.

We have shown that the second move results in no conflict. The proof is complete. □

To extend Theorem 5 to t-nary d-cubes for general d and t remains a challenging problem. Note that a natural extension of Lemma 1 to t-nary d-cubes shows that there exists a 2-move local permutation routing for the t-nary 2-cube for all t.

4. Some remarks

We may also consider an MIN (multistage interconnecting network) version of d-cube, namely, a $(d + 1)$-stage MIN where each stage has 2^d switches (t^d for a t-nary cube) representing the 2^d nodes. Between two adjacent stages there is a link from switch i of the preceding stage to switch j of the succeeding stage if nodes i and j are adjacent in the d-cube. A node is also considered to be adjacent to itself. Thus the link patterns are identical throughout the stages. A d-cube MIN is *rearrangeable* if for an arbitrary matching between switches of the first stage and switches of the last stage, there always exist link-disjoint paths to connect all the pairs. Note that a path $(s_1, s_2, \ldots, s_{d+1})$, where s_i, $i = 1, 2, \ldots, d + 1$, denotes the index of the switch in stage i of the MIN, corresponds to a path from node s_1 to s_{d+1} in the d cube whose ith move is from node s_i to node s_{i+1}. Thus, a packet switching d-cube having a permutation routing with at most d moves would correspond to a circuit-switching rearrangeable d-cube MIN except for the following difference: the d-cube allows two messages to stay put at a node, but in the d-cube MIN this corresponds to two messages competing for the same link (switch i to switch i) which is not allowed. Therefore, rearrangeability for the d-cube MIN implies a permutation routing with at most d moves for the d-cube, while the converse is not true. This difference does not exist for $d = 2$, thus a 2-cube MIN is still rearrangeable. We can easily modify our algorithm to prove this for t-nary 3-cube MIN with base $t = 2, 3$. It should also be noted that the "local" property is preserved in the modification.

A d-cube is circuit-switching rearrangeable if for an arbitrary matching of the 2^d vertices there always exist 2^d edge-disjoint paths connecting the pairs. Szymanski [10] showed that the 2-cube and 3-cube are circuit-switching rearrangeable with minimum-length paths. While routing for circuit switching can certainly be used for packet switching, his routings for the 2-cube and the 3-cube are not local, hence do not imply Corollary 2 and Theorem 4. Szymanski also conjectured that a d-cube is circuit-switching rearrangeable for all d and gave numerical evidences to support the conjecture. Recently, Lubiw [6] showed that this conjecture would be false if all paths are restricted to minimum lengths.

Although Theorem 4 in this paper applies only to binary d-cubes for $d \leqslant 7$, these d-cubes cover some useful practical applications. We also hope that our results will stimulate more research into minimum-move local permutation routing for d-cubes with larger d.

Acknowledgements

We wish to thank the two referees for suggestions leading to a substantial improvement of the exposition.

References

[1] S. Abraham and K. Padmanabhan, Performance of the direct binary N-cube network for multiprocessors, IEEE Trans. Comput. C-38 (7) (1989) 1000–1011.
[2] A. Gottlieb, R. Grishnan, C.P. Kruskal, K.P. McAuliffe, L. Rudolph and M. Snir, The NYU ultracomputer – Designing an MIMD shared memory parallel computer, IEEE Trans. Comput. C-32 (2) (1983) 175–189.
[3] D. Krizance, G. Kaklamanis and T. Tsantilas, Tight bounds for oblivious routing on the hypercube, in: Proceedings of the 2nd Annual ACM Symposium of Parallel Algorithms and Architectures (1990) 31–36.
[4] D.H. Lawrie, Access and alignment of data in an array processor, IEEE Trans. Comput. C-24 (12) (1975) 1145–1155.
[5] J. Lenfant, Permutations of data: a Benes network control algorithm for frequently used permutations, IEEE Trans. Comput. C-27 (7) (1978) 637–647.
[6] A. Lubiw, Counterexample to a conjecture of Szymanski on hypercube routing, Inform. Proc. Lett. 35, (1990) 57–61.
[7] M.V. Heel, A fast algorithm for transposing large multidimensional image data sets, Ultramicroscopy 38 (1991) 75–83.
[8] C.L. Seitz, The cosmic cube. Comm. ACM 28 (1985) 22–23.
[9] D. Steinberg, Notes invariant properties of the shuffle/exchange and a simplified cost-effective version of the omega network, IEEE Trans. Comput. C-32 (5) (1983) 444–450.
[10] T. Szymanski, On the permutation capability of a circuit-switched hypercube, in: Proceedings of the 1989 International Conference on Parallel Processing., IEEE Computer Soc. Press, Silver Spring, MD, Vol. I (1989) 103–110.